98/2021

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F

A2

(11) International Publication Number: WO 95/22791

(43) International Publication Date: 24 August 1995 (24.08.95)

(21) International Application Number: PCT/US95/01779

(22) International Filing Date: 8 February 1995 (08.02.95)

(30) Priority Data:
08/193,383 8 February 1994 (08.02.94) US

(71) Applicant: MERIDIAN SEMICONDUCTOR, INC. [US/US]; Suite 145, 17310 Redhill Avenue, Irvine, CA 92714 (US).

(72) Inventors: WHITTED, Graham, B., III; 25 Hermosa, Irvine, CA 92720 (US). KANE, James, A.; 114 Via Orvieto, Newport Beach, CA 92663 (US). CHANG, Hsiao-Shih; 1134 North Palo Loma Place, Orange, CA 92669 (US).

(74) Agent: SEWELL, Jerry, T.; Knobbe, Martens, Olson and Bear, 16th floor, 620 Newport Center Drive, Newport Beach, CA 92660 (US).

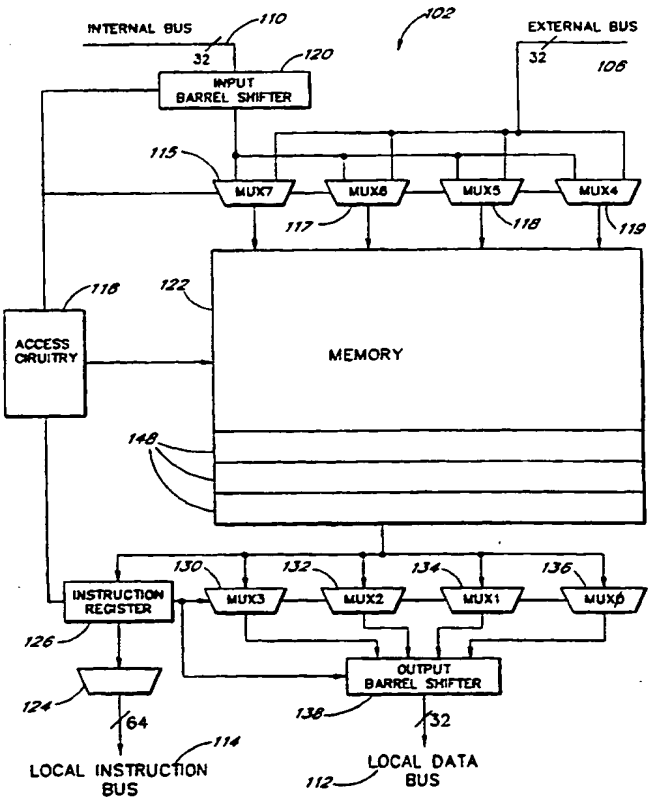
(81) Designated States: CN, JP, KR, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: METHOD AND APPARATUS FOR SINGLE CYCLE CACHE ACCESS ON DOUBLE WORD BOUNDARY CROSS

(57) Abstract

A cache memory system and method control a random access read/write cache memory (122) that stores a plurality of cache data lines (148). The bytes in each cache data line (148) correspond to bytes stored in sequential addresses in a main memory (104). The cache memory (122) is organized to provide storage of a plurality of double words in each cache data line (148). To retain main memory consistency, input circuitry receives and aligns data before storage in the cache memory (122). A separate instruction output path (114) provides access to a plurality of double words in a single cycle. A separate data output path (112) provides access to data that crosses a double word boundary in a single cycle and provides the data as aligned data.



BNSDOCID: WO 9522791A2 I >

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	**GB**	United Kingdom	**MR**	Mauritania
AU	Australia	**GE**	Georgia	**MW**	Malawi
BB	Barbados	**GN**	Guinea	**NE**	Niger
BE	Belgium	**GR**	Greece	**NL**	Netherlands
BF	Burkina Faso	**HU**	Hungary	**NO**	Norway
BG	Bulgaria	**IE**	Ireland	**NZ**	New Zealand
BJ	Benin	**IT**	Italy	**PL**	Poland
BR	Brazil	**JP**	Japan	**PT**	Portugal
BY	Belarus	**KE**	Kenya	**RO**	Romania
CA	Canada	**KG**	Kyrgystan	**RU**	Russian Federation
CF	Central African Republic	**KP**	Democratic People's Republic of Korea	**SD**	Sudan
CG	Congo			**SE**	Sweden
CH	Switzerland	**KR**	Republic of Korea	**SI**	Slovenia
CI	Côte d'Ivoire	**KZ**	Kazakhstan	**SK**	Slovakia
CM	Cameroon	**LI**	Liechtenstein	**SN**	Senegal
CN	China	**LK**	Sri Lanka	**TD**	Chad
CS	Czechoslovakia	**LU**	Luxembourg	**TG**	Togo
CZ	Czech Republic	**LV**	Latvia	**TJ**	Tajikistan
DE	Germany	**MC**	Monaco	**TT**	Trinidad and Tobago
DK	Denmark	**MD**	Republic of Moldova	**UA**	Ukraine
ES	Spain	**MG**	Madagascar	**US**	United States of America
FI	Finland	**ML**	Mali	**UZ**	Uzbekistan
FR	France	**MN**	Mongolia	**VN**	Viet Nam
GA	Gabon				

BNSDOCID: <WO 9522791A2 I >

METHOD AND APPARATUS FOR SINGLE CYCLE CACHE ACCESS ON DOUBLE WORD BOUNDARY CROSS

Background of the Invention

Field of the Invention

The invention relates generally to cache memory and, more particularly, to cache memory that allows single cycle cache access to data that crosses a double word boundary. The invention further relates to a method of using such a cache memory.

Description of the Prior Art

Prior cache memory accesses have been limited to boundaries defined in main memory. The prior art cache memory systems include high speed memory for holding the most recently used information for future reuse by the CPU. Thus, the CPU views the cache as another memory module.

To obtain data and instructions, the CPU has an address bus and a data bus. The address bus accesses the desired main memory location and the data bus communicates the data to the CPU. Since the CPU treats cache as another memory module, the address bus and data bus also access the cache.

To hold down the costs of computers, prior art systems use slower memory for the bulk of a computer's main memory and faster memory in the cache. The cache helps move data between the main memory and the CPU with the least delay. Without the cache, the CPU may sit idle while it waits for retrieval of the requested data. With the cache, however, a computer can keep the data most likely to be requested readily available. The data in the faster cache can be provided to the CPU with a minimum of delay.

When the CPU requests new data, prior art cache memory systems store a copy of the data retrieved from main memory. The next time the CPU requests data, the cache checks to see if data already stored exists in the cache's high speed memory (a hit). If it is, the CPU can access the cache without having to access the slower main memory. The CPU spends less time idling and more time working.

A cache memory system further includes a Data File, a Tag File, a Valid Flag File, and a Least-Recently-Used (LRU) File. The Tag File holds indications of which portions of main memory are stored in the cache. The Valid Flag File indicates which portions of the cache are valid. The Least-Recently-Used (LRU) file defines which portions of the cache to discard. The Data File provides a memory bank for storage of data and instructions fetched from main memory.

BNSDOCID: <WO 9522791A2 1 >

Memory in conventional computers is divided into 8-bit quantities (bytes), 16-bit quantities (words), and 32-bit quantities (double words). In most 32-bit computers, main memory is organized into double word (32-bit) boundaries that correspond to the width of the data bus. To increase processor throughput, prior art cache memory systems are organized to allow access to more memory per cycle than is obtainable on the main memory data path. For example, each address line of the cache memory can hold multiple double words. Thus the cache memory is wider than the main memory data bus.

Often, data samples reside in main memory within a double word boundary. However, a data sample may also cross a double word boundary. For example, data may cross a double word boundary in main memory such that a portion of the data sample resides in a first double word and a portion resides in an adjacent double word. To retrieve such a data sample in prior art systems, two read cycles are required: one cycle to read from the data within the first double word boundary; and one cycle to read the data in the adjacent double word boundary.

Although prior art cache memory systems can access multiple double words, the prior art systems still operate on double word boundaries in order to maintain consistency with main memory. Therefore, these prior art cache memory systems suffer from several drawbacks. For example, two read cycles are necessary to retrieve data that crosses a double word boundary. Also, data that crosses a double word boundary cannot be stored into the cache in a single cycle. Furthermore, prior art cache memories typically do not provide alignment circuitry to allow single cycle accesses to cache memory while maintaining consistency with main memory.

In addition to data, some cache memory systems also allow storage of instructions. To enhance performance, processors often decode instruction codes in a pipeline fashion. In exemplary prior art, an instruction unit in the CPU processes instruction codes. To enhance instruction decoding and processing, prior art systems employ a wide instruction bus to move instructions between the cache and the instruction unit. A wide instruction bus can take advantage of a simultaneous cache access to multiple double words.

However, prior art cache systems fail to differentiate between instruction and data cache accesses that cross double word boundaries. When data crosses a double word boundary, cache circuitry needs to provide proper alignment, whereas instructions that cross double word boundaries do not need alignment. Furthermore, prior art cache memories do not provide circuitry to allow direct access for the wider instruction bus, and an aligned access for a separate data bus.

Summary of the Invention

In accordance with the present invention, a cache memory alignment system for single cycle access to data that crosses double word boundaries is disclosed. The subject invention comprises a

PNSDOCID: <WO 9522791A2 I >

microprocessor having a central processing unit (CPU) and an on-chip cache that enhances the microprocessor processing speed.

By providing separate paths for instructions and data, the cache access circuitry allows single cycle access to larger instruction segments and to data that crosses double word boundaries. The cache memory uses a plurality of random access memories (RAMs) to provide a memory bank 16 bytes wide and 512 lines high. The reliance on a plurality of RAMs provides the further benefit of accessing a byte, word, or double word for data, and multiple double words for instructions. Barrel shifters and multiplexers are used to eliminate unwanted read and write delays.

The 16 RAMs share the same address lines. The memory accessed by the address lines is known as a cache data line. In the preferred embodiment, nine address lines access 512 cache data lines. When the CPU initiates an access, the Tag File, Valid Flag File and LRU File use the physical address lines on the address bus to enable a corresponding cache data line.

For instructions, the access circuitry allows single cycle access of eight bytes of instruction information from a cache data line. For data, the access circuitry allows single cycle access to data that crosses a double word boundary within a cache data line. The advantage of this design is higher total performance. All cache data accesses take one cycle. All cache instruction accesses provide up to 16 bytes of data in two cycles.

Accordingly, the present invention includes an address bus, an internal bus, a local data bus, a local instruction bus, and an external data bus. The external data bus, the internal bus, and local data bus are 32-bits wide. The local instruction bus is 64-bits wide. The internal bus interconnects various CPU registers to the cache. The external data bus interconnects the CPU, cache and main memory. The internal bus allows transfers from the CPU to cache. The access circuitry includes an input multiplexer, an input barrel shifter, an output instruction multiplexer, output data multiplexers, and an output barrel shifter.

Data stored in the cache RAMs correspond to data stored in sequential addresses in a main memory. In the preferred embodiment, the main memory is organized on double word boundaries. The access circuitry stores each double word from main memory into a corresponding double word in the cache RAMs.

A factor that enhances the cache's speed and flexibility is the use of an input barrel shifter. The input barrel shifter holds a double word and shifts left to align data before storage into the cache RAMs. The byte ordering of the barrel shifter corresponds to the byte ordering of double words in the cache RAMs.

The output circuitry also enhances the cache's speed and flexibility. The instruction output path includes an instruction register and an instruction multiplexer. The instruction register receives and stores

SUBSTITUTE SHEET (RULE 26)

the entire cache data line. The instruction multiplexer connects to the output of the instruction register and selects eight bytes for output to a local instruction bus. Thus, for instructions, the apparatus of the present invention allows the retrieval of 8 bytes in a single cycle and 16 bytes in two cycles.

The data output path includes a plurality of multiplexers and an output barrel shifter. The plurality of multiplexers, and output barrel shifter allow retrieval of a four-byte sequence in a cache data line in a single cycle.

In accordance with the method of the invention, a byte, word, or double word is fetched from main memory. A select line on the input multiplexer is asserted to allow the external data bus to load the input barrel shifter. If data is sent from the CPU, a select line on the input multiplexer is asserted to allow the internal bus to load the input barrel shifter.

Access circuitry decodes the address and size of the data sample and selects a particular cache data line. The access circuitry also determines alignment of the double word in the cache data line. If alignment is necessary, the access circuitry commands the input barrel shifter to shift left. The access circuitry also asserts the corresponding write enables of the cache data line in order to load the data from the input barrel shifter into the cache data line.

To retrieve instructions from the cache, access circuitry decodes the address and selects a particular cache data line. The contents of the cache data line are loaded into the cache instruction register. Physical address line 3 drives the select line of the instruction multiplexer. If unasserted, the select line commands the instruction multiplexer to select the first eight bytes of the cache instruction register. If asserted, the select line commands the instruction multiplexer to select the second eight bytes of the cache register. The instruction multiplexer outputs to the 64-bit internal instruction bus.

To retrieve a byte, word, or double word of data from cache, access circuitry decodes the address and size of the data sample and selects a particular cache data line. The access circuitry also asserts the select lines to each data multiplexer in order to retrieve four consecutive bytes in the cache data line. The output of the data multiplexers loads the barrel shifter. If alignment is necessary, the access circuitry commands the output barrel shifter to shift right. After shifting, the barrel shifter outputs the data to an internal data bus.

In a preferred embodiment of the present invention, a cache memory for storing data accessible by a central processing unit includes a plurality of random access read/write memories (RAMs). The plurality of RAMs have a plurality of address inputs, a plurality of RAM data inputs, and a plurality of RAM data outputs. The plurality of RAMs store a plurality of cache data lines. The plurality of double words in each cache data line correspond to double words stored in sequential addresses in a main memory.

BNSDOCID: <WO 9522791A2 I >

In another aspect of the invention, input circuitry receives and aligns input data before storage in an addressed cache data line. For example, the input data may include multiple bytes where a least significant byte is stored in a non-least significant byte location of a first double word in the addressed cache data line, and where the input data's most significant byte is stored in a non-most significant byte location of a second adjacent double word in the addressed cache data line.

To store such a data sample in a single cycle, the input circuitry selectively shifts the input data to position the least significant byte in the non-least significant byte location and the most significant byte in the non-most significant byte position. The shifting of the data sample before storage into the cache data line retains main memory consistency.

A further aspect of the invention includes a data output path coupled to the RAM data outputs. The data output path receives output data from an enabled cache data line and aligns the output data in a single access cycle. For example, the output data may include multiple bytes with a least significant byte stored in a non-least significant byte location of a first double word of the enabled cache data line and include a most significant byte stored in a non-most significant byte location of a second adjacent double word in the enabled cache data line. The output circuitry selectively shifts the output data to position the least significant byte to the least significant byte position within the multiple byte output.

Yet another aspect of the invention includes an instruction output path coupled to said RAM data outputs that selects and outputs a plurality of instruction bytes from a plurality of double words of an addressed cache data line.

These and other aspects, advantages and novel features of the invention will become apparent upon reading the following detailed description of the invention and upon reference to the accompanying drawings in which:

Brief Description of the Drawings

Fig. 1 is a block diagram of a microprocessor system comprising a central processing unit (CPU), a cache, and an external main memory.

Fig. 2 is a block diagram of a cache data file showing an internal bus, external data bus, access circuitry, an input barrel shifter, input multiplexers, memory, an instruction register, an instruction multiplexer, a plurality of data output multiplexers, and an output barrel shifter.

Fig. 3 is a block diagram of a RAM.

Fig. 4 is a table of cache memory locations illustrating the RAMs organized into four double words.

Fig. 5 is a block diagram of the access circuitry.

Fig. 6 is a schematic of the input barrel shifter connected to the RAMs.

Fig. 7 is a truth table of input barrel shifter lines BARREL_IN1 and BARREL_IN0.

Fig. 8a and 8b is a truth table of alignment register values, size values, and bytes in a cache data line.

FIG. 9 is a block diagram of the access circuitry, RAMs, data multiplexers and the output barrel shifter.

FIG. 10 is a truth table of the alignment register and corresponding data multiplexer outputs.

Fig. 11 is a truth table of output barrel shifter lines BARREL_OUT1 and BARREL_OUT0.

Fig. 12 is a block diagram of the input circuitry and a data sample W, X, Y, and Z that crosses a double word boundary.

FIG. 13 is a timing diagram of a fetch of four data words from main memory.

Fig. 14 is a timing diagram of a data read cycle in the cache memory of the present invention.

Fig. 15 is a timing diagram of two instruction fetch cycles in the cache memory system of the present invention.

Fig. 16 is a block diagram of the output circuitry and a data sample W, X, Y, and Z that crosses a double word boundary.

Detailed Description of the Preferred Embodiment

FIG. 1, illustrates a microprocessor having an on-chip cache. The microprocessor comprises a central processing unit (CPU) 100, and a cache 102. The on-chip cache 102 connects to a main memory 104 via an external data bus 106 and an address bus 108. The CPU 100 accesses the cache via the address bus 108 and the internal bus 110. In addition, the CPU accesses the cache memory via a local data bus 112, and a local instruction bus 114.

In the preferred embodiment, the external data bus 106, the address bus 108, the internal bus 110, and local data bus 112 are 32-bits wide. The local instruction bus 114 is 64-bits wide. The internal bus 110 connects various CPU registers to the cache. The external data bus connects the cache 102 and main memory 104. The internal bus 110 allows high speed transfers between the CPU 100 and cache 102.

FIG. 2 illustrates a block diagram of the preferred embodiment of the present invention. A cache memory, or Data File 102, stores data and instructions retrieved from main memory. The Data File 102 includes access circuitry 116, an input barrel shifter 120, input multiplexers 115, 117, 118, 119, memory 122, an instruction register 126, an instruction multiplexer 124, data output multiplexers 130, 132, 134, 136, and an output barrel shifter 138.

The memory 122 organizes a plurality of RAMs into a memory module. As shown in FIG. 3, each RAM 140 is provided with a plurality of address lines 142, and a set of eight data lines (one byte)

BNSDOCID: <WO 9522791A2 I >

144. In the preferred embodiment, nine address lines access 512 memory locations. A write enable (WE) 146 is provided to strobe data from the set of data lines 144 into the RAM 140. The data is strobed into the RAM 140 at a location determined by the access circuitry (not shown). Thus, the plurality of address lines access 512 locations in the RAM 140, each location holds eight bits (one byte).

As shown by a block diagram in FIG. 4, the Data File includes 16 RAMs 140. Conceptionally, the cache memory is a two dimensional matrix, 512 rows (lines) high and 16 columns (bytes) wide. The organization of the 16 RAMs allows the cache to access 16 bytes of data in a single cycle. The 16 RAMs share the same address lines 142. When a particular address is asserted, a byte in each RAM is accessed to provide 16 bytes of memory. The 16 bytes of memory that correspond to a particular address are known as a cache data line 148. Since the address lines access 512 locations, the cache has 512 cache data lines 148. The 16 bytes of each cache data line 148 are numbered from 0 to 15 (where zero is the least significant byte and 15 is the most significant byte).

The cache further includes a Tag File, a Valid Flag File, and a Least-Recently-Used (LRU) File (not shown). The Tag File, Valid Flag File, and LRU File are well known in the prior art. The Tag File holds indications of which portions of main memory are stored in the cache. The Valid Flag File indicates which portions of the cache are valid. The LRU file defines which portions of the cache to discard.

When the CPU initiates an access, the Tag File, Valid Flag File and LRU File use the physical address lines on the address bus to enable a corresponding cache data line 148. The Tag File contains four sets of 128x21 RAMs and 21-bit comparators. The Tag File stores bits 31:10 of the physical addresses of the data stored in the cache. The Valid Flag File has four 128x1 RAMs for storing the valid bits that determine if valid data exists at a particular location in the cache. The LRU File (Least Recently Used) includes three 128x1 RAMs. Once the cache memory is loaded, the LRU defines what locations to discard. No further discussion of the Tag File, Valid Flag File, or LRU File is made herein as one who is skilled in the relevant technology will understand their function.

Memory in conventional computers is divided up into 8-bit quantities (bytes), 16-bit quantities (words), and 32-bit quantities (double words). In most 32-bit computers, main memory is organized into double word (4 byte) boundaries. To maintain consistency with main memory, the 16 bytes in each cache data line is further partitioned into four double words. Each cache double word corresponds to a double word in main memory. When the cache memory is viewed as a matrix, each row (cache data line) has 16 bytes, the 16 bytes are further organized into four large columns (double words).

As illustrated in FIG. 4, the 16 bytes of each cache data line 148 are numbered from 0 to 15 (where zero is the least significant byte and 15 is the most significant byte). The first double word occupies bytes 0-3, the second double word occupies bytes 4-7, the third double word occupies bytes 8-11 and the fourth double word occupies bytes 12-15 of each cache data line 148.

BNSDOCID: <WO 9522791A2 >

Referring to FIG. 1, when the CPU 100 initiates a fetch from main memory 104, it also initiates a cache access. If the requested data is not in the cache 102 (a miss), then a fetch from main memory 104 is generated on the external data bus 106. The fetch from main memory 104 retrieves four consecutive double words. The four double words (16 bytes) from main memory are copied into the selected cache data line. The CPU generates the address lines that determine which 16 consecutive bytes to retrieve from main memory.

As shown in FIG. 2 and 5, the access circuitry 116 controls the input barrel shifter 120, the input multiplexers 115, 117, 118, 119, memory (RAMs) 122, the instruction register 126, the instruction multiplexer 124, the data output multiplexers 130, 132, 134, 136 and the output barrel shifter 138. Referring to FIG. 5, the access circuitry includes an alignment register 150, a size register 152, and control logic 154. The alignment register 150 receives physical address lines 3-0.

The size register 152 identifies how many bytes to access in the selected cache data line 148. Since it is possible to access a byte, word, or double word, the alignment register 150 identifies the lowest order byte, and the size register 152 identifies any additional bytes. The control logic 154, the size register 152, and the alignment register 150 determine which write enables 146 of the cache data line 148 are asserted.

Turning to the input circuitry illustrated in FIG. 2, the input barrel shifter 120, input multiplexers 115, 117, 118, 119, and access circuitry 116 load the cache memory (RAMs) 122. The input barrel shifter 120 connects to the internal bus 110. The input multiplexers 115, 117, 118, and 119 connect to the external data bus 106 and the outputs of the input barrel shifter 120. The internal bus 110 and external data bus 106 are 32-bits wide and carry both data and instructions.

The input barrel shifter 120 transfers the contents of the internal data bus 110 to the input multiplexers 115, 117, 118, 119. The input multiplexers 115, 117, 118, 119 are two-to-one multiplexers that select data from the external data bus 106 or the input barrel shifter 120. The outputs of input multiplexers 115, 117, 118, 119 connect to the cache memory 122. Referring to FIG. 6, in the preferred embodiment, the input barrel shifter 120, holds a double word (4 bytes) and can shift each byte left with the most significant byte wrapping around to the least significant byte location. The wrap around shift left capability of the barrel shifter 120 allows alignment of data before storage into the enabled cache data line 148. The shift left moves each byte in the barrel shifter 120 left one byte. The most significant byte wraps around to the least significant byte.

Each byte of the input barrel shifter 120 connects to one of the input multiplexers 115, 117, 118, 119. In FIG. 2 the four input multiplexers are also referred to as MUX4 119, MUX5 118, MUX6 117, and MUX7 115. The first byte (least significant byte) of the barrel shifter 120 connects to MUX4 119. The second byte of the barrel shifter 120 connects to MUX5 118. The third byte of the barrel

RNSDOCID: WO 9522791A2 I >

shifter connects to MUX6 117. The fourth byte of the barrel shifter (most significant byte) connects to MUX7 115.

In FIG. 6, the letters W, X, Y, and Z represent the byte ordering of the input barrel shifter 120 that corresponds to bytes in the cache data line 148. The lowest order byte of the barrel shifter (Z) connects MUX4 119. The highest order byte of the barrel shifter (W) connects to MUX7 115. Accordingly, the middle two bytes (X and Y) connect to MUX6 117, and MUX5 118.

In a similar fashion, each byte of the external data bus 106 connects to the input multiplexers 115, 117, 118, 119. The first byte (least significant byte) of the external data bus 106 connects to MUX4 119. The second byte of the external data bus 106 connects to MUX5 118. The third byte of the external data bus 106 connects to MUX6 117. The fourth byte of the external data bus (most significant byte) connects to MUX7 115.

The access circuitry 116 controls the input multiplexers 115, 117, 118, 119 with a select line that selects either the external data bus 106 or the internal data bus 110. Each of the input multiplexers 115, 117, 118, and 119 connect to four bytes in the cache data line 148. Conceptionally the input multiplexers correspond to the byte ordering of each double word in the cache data line 148.

Thus the output of MUX4 119 connects to bytes 0, 4, 8, and 12 in the Cache Data Line. The output of MUX5 118 connects to bytes 1, 5, 9, and 13. The output of MUX6 117 connects to bytes 2, 6, 10, and 14. Finally, the output of MUX7 115 connects to bytes 3, 7, 11, and 15.

To provide flexibility, the input barrel shifter 120 can align bytes for storage into the cache data line 148. To properly align the bytes, the output barrel shifter shifts data left. Two control lines, BARREL_IN1 and BARREL_IN0 determine how far to shift data left. FIG. 7 illustrates a truth table for the control lines BARREL_IN1 and BARREL_IN0. It should be understood that because the four byte outputs of the barrel shifter are replicated for each double word, it s not necessary to shift left more than three bytes.

Referring to FIG. 5, to write a byte, the control logic 154 asserts the write enable identified by the alignment register 150. To write a word the control logic 154 asserts the write enable identified in the alignment register 150 and the next higher order byte. To write a double word, the control logic 154 asserts the write enable identified by the alignment register 150 and the write enables of the next three higher order bytes. FIGs. 8a and 8b show a table of the alignment register values (hexadecimal), the byte, word or double word size, and the corresponding write enable asserted by the access circuitry.

Referring to FIGs. 2, 5, 8a, and 8b, after the input multiplexers select either the external data bus 106, or the input barrel shifter 120, the control logic 154 asserts the appropriate write enables 146 identified by the alignment register 150. For example, to write a word at byte location zero in a cache data line 148 (see line 2 in FIG. 8a), the control logic 154 asserts the write enable identified in the

BNSDOCID: <WO__9522791A2_I_>

alignment register 150. The alignment register 150 identifies byte zero in the cache data line 148. The size register 152 identifies that the data sample is two bytes (a word), thus the control logic 154 also asserts the write enable 146 to byte one in the cache data line 148.

To write a double word, the control logic 154 asserts the write enable identified by the alignment register 150 and the write enables 146 of the next three higher order bytes. For example, to write a double word at byte location eight in a cache data line 148 (see first line in FIG. 8b), the alignment register 150 identifies byte eight, and the size register 152 identifies a double word. The control logic 154 asserts the write enables 146 to bytes eight, nine, ten, and eleven in the cache data line 148.

Turning now to the output circuitry as shown in FIG. 2, once data or instructions are stored into the cache data line 148, the CPU can access the cache data line 148 in a single cycle. For this purpose, the cache includes a separate output path for instructions and data.

The instruction output path includes an instruction register 126, and an instruction multiplexer 124. In the preferred embodiment, the instruction register 126 is 16 bytes wide and stores the contents of an enabled cache data line 148. The instruction multiplexer 124 is eight bytes (64-bits) wide and outputs to the local instruction bus 114 which is also eight bytes (64-bits) wide. For instruction fetches, the output circuitry transmits the entire cache data line 148 in two cycles. The first cycle stores the entire cache data line 148 into the instruction register 126 and the instruction multiplexer 124 selects the first eight bytes of the cache data line. The second cycle commands the instruction multiplexer 124 to select the second eight bytes in the instruction register 126.

Physical address line 3 drives the select line of the instruction multiplexer 124. If address line 3 is not asserted, the instruction multiplexer 124 selects the first eight bytes (bytes 0-7) of the instruction register 126. If address line 3 is asserted, the instruction multiplexer 124 selects the second eight bytes (bytes 8-15) of the instruction register 126.

In order to retrieve data, the data output path includes the four data output multiplexers 130, 132, 134, 136, and the output barrel shifter 138. The data output multiplexers are designated as MUX3 130, MUX2 132, MUX1 134, and MUX0 136. The four data output multiplexers 130, 132, 134, 136 and output barrel shifter 138 allow access to any four byte sequence in the 16 byte cache data line.

Each multiplexer connects to four bytes in the cache data line 148 as shown in FIG. 9. The letters W, X, Y, and Z show the correspondence of the bytes in the cache data line 148 with the data output multiplexers 130, 132, 134, and 136. MUX3 130 connects to the fourth byte (highest order byte W) of each double word. MUX0 136 connects to the first byte (lowest order byte Z) of each double word. MUX2 132 and MUX1 134 correspond to the third and second bytes (X and Y) of each double word in like manner. Therefore, MUX3 130 connects to bytes 15, 11, 7 and 3. MUX2 132 connects to

BNSDOCID: <WO 9522791A2 >

bytes 14, 10, 6 and 2. MUX1 134 connects to bytes 13, 9, 5 and 1. MUX0 136 connects to bytes 12, 8, 4, and 0. FIG. 10 illustrates a truth table of the alignment register values and corresponding cache data line bytes selected by MUX3 130, MUX2 132, MUX1 134, and MUX0 136.

Each multiplexer has two select lines that control which byte to select. Select lines MUX3_SEL1 and MUX3_SEL0 control MUX3 130. Select lines MUX2_SEL1, and MUX2_SEL0 control MUX2 132. Select lines MUX1_SEL1, and MUX1_SEL0 control MUX1 134. Select lines MUX0_SEL1, and MUX0_SEL0 control MUX0 136. Table 1 illustrates a truth table for the select lines of each data multiplexer.

MUX3 Selects from bytes: 15, 11, 7, 3			MUX2 Selects from bytes: 14, 10, 6, 2			MUX1 Selects from bytes: 13, 9, 5, 1			MUX0 Selects from bytes: 12, 8, 4, 0		
SEL1	SEL0	Byte	SEL1	SEL0	Byte	SEL1	SEL0	Byte	SEL1	SEL0	Byte
0	0	3	0	0	2	0	0	1	0	0	0
0	1	7	0	1	6	0	1	5	0	1	4
1	0	11	1	0	10	1	0	9	1	0	8
1	1	15	1	1	14	1	1	13	1	1	12

Table 1

Since each multiplexer is hard-wired to particular bytes in the cache data line 148, the bytes selected by the data output multiplexers 130, 132, 134, 136 may need alignment. To properly align the bytes, the output barrel shifter 138 shifts data right. When shifting right, the output barrel shifter 138 wraps the least significant byte around to the most significant byte location. Two control lines, BARREL_OUT1 and BARREL_OUT0 determine how far to shift data right. The access circuitry 116 generates control lines BARREL_OUT1 and BARREL_OUT0. FIG. 11 shows a truth table for the control lines BARREL_OUT1 and BARREL_OUT0.

In accordance with the method of the invention, the cache allows accesses of bytes, words and double words. As the CPU 100 requests new data, the cache stores a copy of the data retrieved from main memory. Referring to FIGs. 1 2, and 6, the access circuitry decodes the address and size of the data sample and enables the appropriate cache data line 148. The access circuitry 116 also directs the input multiplexers 115, 117, 118, 119 to select the external data bus 106. The byte ordering on the external data bus 106 will correspond to the bytes addressed in main memory. Since data retrieved from the external data bus 106 aligns within double word boundaries, no alignment is performed.

If data is sent from the CPU 100, the access circuitry 116 directs the input multiplexers 115, 117, 118, 119 to select the output of input barrel shifter 120. The access circuitry 116 also determines alignment of the double word in the cache data line 148. If alignment is necessary, the access circuitry

SUBSTITUTE SHEET (RULE 26)

116 commands the input barrel shifter 120 to shift left with BARREL_IN1 and BARREL_IN0. During a shift left, the input barrel shifter 120 wraps the most significant byte around to the least significant byte location. The access circuitry 116 asserts the corresponding write enables 146 of the cache data line 148 in order to load the data from the input barrel shifter 120 into the cache data line 148.

FIG. 12 shows a data sample in main memory that crosses a double word boundary. The data sample comprises four bytes of memory, W, X, Y, and Z. Bytes Y and Z reside in the high order portion of a first double word 156, and bytes W and X reside in the lower order portion of an adjacent double word 158.

To retrieve such a data sample in main memory, two cycles are required. Referring to the timing diagrams in FIGs. 13 and 14, and the block diagram in FIG. 11, the CPU 100 first issues an access command. If the data is not in the cache, the CPU 100 reads from the main memory 104. The access circuitry 116 directs the input multiplexers 115, 117, 118, 119 to select the external data bus 106. As shown by the timing diagram in FIG. 13, during one cycle, bytes W and X are transferred to bytes 5, and 4 in the cache data line 148. In the other cycle, bytes Y, and Z are transferred to bytes 3 and 2 in the cache data line. Two additional read cycles retrieve the third and fourth double words from main memory to load the third and fourth double words in the cache data line.

After processing, the CPU 100 attempts to store sample W, X, Y, and Z. However, instead of the two cycles required to access main memory 104, the CPU 100 can access the cache data line 148 in one cycle, as shown by the timing diagram in FIG. 14. The CPU 100 initiates an access (write) cycle. The input barrel shifter 120 loads data sample W, X, Y, and Z from the internal bus 110. The access circuitry 116 directs the input multiplexers 115, 117, 118, 119 to select input barrel shifter 120.

Since each byte of the input barrel shifter 120 is hard-wired via the input multiplexers 115, 117, 118, 119 to specific byte locations in the cache data line 148, the input barrel shifter 120 must align the bytes before storage into the cache data line 148. To properly align the data sample, the following will occur. As illustrated in FIG. 12, the access circuitry asserts BARREL_IN1 to command the barrel shifter to rotate W, X, Y, and Z two bytes left, so that Y, Z, W, and X appear in the barrel shifter in that order. Furthermore, the access circuitry 116 asserts the write enables to bytes 5, 4, 3, and 2 in the cache data line 148. FIG. 8a shows that bytes 5, 4, 3, and 2 are enabled when alignment register holds a two and the size register holds a double word.

Input multiplexer 119 passes the first byte (lowest order byte) in the input barrel shifter 120 (now X) to byte 4 in the cache data line 148. Input multiplexer 118 passes the second byte in the input barrel shifter 120 (now W) to byte 5. Input multiplexer 117 passes the third byte in the barrel shifter (now Z) to byte 2. Finally, input multiplexer 115 passes the fourth byte in the input barrel shifter 120

(now Y) to byte 3. Within in one access cycle, data sample W, X, Y, and Z resides in the cache data line at bytes 5, 4, 3, and 2.

As explained, the access circuitry 116 allows access of bytes, words, and double words within the cache data line. However, a data sample may cross a cache boundary. For example, a data sample W, X, Y, and Z resides in cache such that bytes Y and Z reside in the fourth double word of a first cache data line 148, and bytes W and X reside in the first double word of a second cache data line 148. The access circuitry 116 will access bytes Y and Z in the first cache data line 148 in one cycle, and access W and X in the second cache data line 148 in a second cycle. For data that crosses a cache boundary, the cache will access the data in two cycles.

The timing diagram in FIG. 15 shows a fetch of instructions in the present invention. The CPU issues an access command and retrieves four double words of instructions from main memory in four cycles. The four double words are also loaded into an enabled cache data line 148.

When the CPU issues an access for the same instructions, the cache determines that the instructions reside in cache memory (a hit). The access circuitry 116 enables the appropriate cache data line 148. The 16 bytes in the enabled cache data line 148 are loaded into the instruction register 126. In one cycle, the instruction multiplexer 124 selects the first and second double words of the instruction register 126 and outputs to the local instruction bus 114. In a second cycle, the instruction multiplexer 124 selects the second and third double words from the instruction register 126 and outputs to the local instruction bus. Therefore, the cache of the present invention can fetch eight instruction bytes in a single cycle and 16 instruction bytes in two cycles.

For instructions, the following example illustrates the operation of the cache. Sixteen instruction bytes K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, and Z reside in a cache data line. The CPU initiates an instruction fetch to retrieve bytes K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, and Z. The cache determines that the instruction information resides in the cache RAMs. The access circuitry enables the appropriate cache data line and loads the instruction register. The instruction multiplexer selects the lower eight bytes from the instruction register and outputs S, T, U, V, W, X, Y, and Z onto the local instruction bus.

In a second fetch cycle, the instruction multiplexer selects the upper eight bytes of the cache data line. The instruction multiplexer outputs the eight upper bytes of the instruction register K, L, M, N, O, P, Q, and R on to the local instruction bus. Thus, the CPU can retrieve 16 bytes of instruction information in two fetch cycles.

Turning to the data output path, the cache of the preferred embodiment can retrieve a byte, word, or double word. The access circuitry 116 decodes the address and size of the data sample and selects a particular cache data line 148. The access circuitry 116 also asserts the select lines to each

data output multiplexer 130, 132, 134, 136 in order to retrieve four consecutive bytes in the cache data line 148. The data output multiplexers 130, 132, 134, 136 connect to the input of the output barrel shifter 138. If alignment is necessary, the access circuitry 116 commands the output barrel shifter 138 to shift right. After shifting, the output barrel shifter 138 outputs the data to the local data bus 112.

To retrieve data, the access circuitry can access data sizes of a byte, word, or double word in a single cycle. Table 2 illustrates an example of four bytes of data aligned within a double word boundary.

Cache Data Line

Fourth Double Word				Third Double Word				Second Double Word				First Double Word			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												W	X	Y	Z

Table 2

The four bytes of data are designated as W, X, Y, and Z. W is the highest order byte and Z is the lowest order byte. Bytes W, X, Y, and Z exist in bytes 3, 2, 1, and 0 of the cache data line. Conceptionally, bytes W, X, Y, and Z reside in the first double word of the cache data line.

To retrieve W, X, Y, and Z, the CPU initiates a read data cycle. The cache determines that W, X, Y, and Z reside in the cache RAMs (a hit). As illustrated in FIG. 2, the access circuitry enables the appropriate cache data line 148. The access circuitry 116 asserts the select lines of the data multiplexers. The data output multiplexers, MUX3 130, MUX2 132, MUX1 134, and MUX0 136 select data bytes 3, 2, 1, and 0 in the cache data line as shown in Table 3.

Alignment Register	Size	MUX3 (byte)	MUX2 (byte)	MUX1 (byte)	MUX0 (byte)
0	double word	3 W	2 X	1 Y	0 Z

Table 3

The data output multiplexers 130, 132, 134, 136 load the output barrel shifter 138 with W, X, Y, and Z. Since W, X, Y, and Z align within a double word boundary no shifting is necessary. Thus the access circuitry does not assert BARREL_OUT1 or BARREL_OUT0, and the output barrel shifter 138 loads W, X, Y, and Z onto the internal data bus.

Since the four data output multiplexers MUX3 130, MUX2 132, MUX1 134 and MUX0 136 also allow access of data that crosses a double word boundary, the data may need alignment before sending it

RNSDOCID: WO 9522791A2 I >

to the CPU 100. As an example, four bytes that cross a double word boundary are designated as W, X, Y, and Z as shown in FIG. 16.

W is the highest order byte and Z is the lowest order byte. Bytes W and X reside in the cache data line 148 at bytes 13 and 12. Bytes Y and Z reside in bytes 11 and 10. Bytes W and X reside in the fourth double word, while bytes Y and Z reside in the third double word. To retrieve W, X, Y, and Z the CPU 100 initiates a read cycle. The cache determines that W, X, Y, and Z reside in the cache RAMs.

The access circuitry 116 enables the appropriate cache data line. FIG. 16 shows that the access circuity also asserts the select lines on the data output multiplexers 130, 132, 134, 136 to select data bytes 13, 12, 11, and 10.

Because the data output multiplexers 130, 132, 134, 136 are hard-wired to specific bytes in the cache data line 148, the access circuitry 116 directs MUX3 130 to select byte 12, MUX2 132 to select byte 11, MUX1 134 to select byte 14, and MUX0 136 to select byte 13. As a result, the byte ordering differs from the original sample. The data multiplexers load Y, Z, W, and X into the output barrel shifter 138. The access circuitry 116 asserts BARREL_OUT1, and the barrel shifter shifts right two bytes to produce W, X, Y, and Z. Once aligned, the output barrel shifter 138 outputs W, X, Y, and Z onto the local data bus 112.

Table 4 illustrates another example where four bytes of data cross a double word boundary. The four bytes are designated as W, X, Y, and Z. W is the highest order byte and Z is the lowest order byte.

Cache Data Line

Fourth Double Word				Third Double Word				Second Double Word				First Double Word			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			W	X	Y	Z									

Table 4

To retrieve W, X, Y, and Z, the CPU 100 initiates a read data cycle. The cache 102 determines that W, X, Y, and Z reside in the cache RAMs. The access circuitry 116 enables the appropriate cache data line 148. Because the multiplexers are hard-wired to specific bytes in the cache data line, the access circuitry 116 asserts the select lines on the data multiplexers so that MUX3 130 selects byte 11, MUX2 132 selects byte 10, MUX1 134 selects byte 9, and MUX0 136 selects byte 12 as shown in Table 5.

As a result, the byte ordering differs from the original sample. The data output multiplexers 130, 132, 134, 136 load X, Y, Z, and W into the output barrel shifter 138. The access circuitry 116 asserts BARREL_OUT1, and BARREL_OUT0. The output barrel shifter 138 shifts right three bytes to

produce W, X, Y, and Z. Once aligned, the output barrel shifter 138 outputs W, X, Y, and Z onto the local data bus.

Alignment Register	Size	MUX3 (byte)	MUX2 (byte)	MUX1 (byte)	MUX0 (byte)
9	double word	11 X	10 Y	9 Z	12 W

Table 5

From this detailed description, taken in conjunction with the appended drawings, the advantages of the present invention will be readily understood by one who is skilled in the relevant technology. The present apparatus and method provides speed a flexibility by providing alignment, storage, and retrieval of data that crosses double word boundaries in a single access cycle. In addition, the separate instruction output path allows access to an entire cache data line 148 in two cycles.

While the above detailed description has shown, described and pointed out the fundamental novel features of the invention as applied to various embodiments, it will be understood that various omissions and substitutions and changes in the form and details of the illustrated device may be made by those skilled in the art, without departing from the spirit of the invention.

RNSDOCID: <WO 9522791A2 I >

WHAT IS CLAIMED IS:

1. A cache memory system for storing data to be accessed by a central processing unit, said cache memory system comprising:

a random access read/write memory (RAM) for storing a plurality of cache data lines, said RAM having a plurality of address inputs, a plurality of RAM data inputs, and a plurality of RAM data outputs, each cache data line holding a plurality of double words;

input circuitry coupled to said plurality of RAM data inputs, said input data circuitry receiving input data from said central processing unit and aligning said input data to be stored in an addressed cache data line, said input data including multiple bytes with a least significant byte to be stored in a non-least significant byte location of a first double word in said cache data line, and including a most significant byte to be stored in a non-most significant byte location of a second adjacent double word in said cache data line, said input circuitry selectively shifting said input data to position said least significant byte in said non-least significant byte location, and to position said most significant byte in said non-most significant byte position; and

a data output path coupled to said RAM data outputs, said data output path receiving output data from an enabled cache data line and aligning said output data, said output data including multiple bytes with a least significant byte stored in a non-least significant byte location of a first double word in said cache data line and including a most significant byte stored in a non-most significant byte location of a second adjacent double word in said cache data line, said output circuitry selectively shifting said output data to position said least significant byte to the least significant byte position of said multiple byte output.

2. A cache memory system as defined in Claim 1, wherein said input circuitry further includes an input barrel shifter coupled to said RAM data inputs to shift said input data.

3. A cache memory system as defined in Claim 2, wherein said input circuitry further includes an input multiplexer coupled to an internal bus and an external data bus, the outputs of said input multiplexer coupled to said barrel shifter, said input multiplexer selectively communicating said input data from one of said internal bus or said external bus to a multiplexer output.

4. A cache memory system as defined in Claim 1, wherein said input circuitry shifts and stores said input data in a single access cycle.

ENSDOCID: <WO 9522791A2 I >

5. A cache memory system as defined in Claim 1, wherein said data output path includes a plurality of data multiplexers coupled to said plurality of RAM data outputs for receiving and selectively transferring said output data.

6. A cache memory system as defined in Claim 5, wherein said data output path includes an output barrel shifter coupled to said plurality of data multiplexers, said output barrel shifter receiving and shifting said output data.

7. A cache memory system as defined in Claim 1, wherein said data output path retrieves and aligns said output data in a single access cycle.

8. A cache memory system for storing data to be accessed by a central processing unit, said cache memory system comprising:

a random access read/write memory (RAM) for storing a plurality of cache data lines, said RAM having a plurality of address inputs, a plurality of RAM data inputs, and a plurality of RAM data outputs, each cache data line holding a plurality of double words;

input circuitry coupled to said plurality of RAM data inputs, said input data circuitry receiving input data and aligning said input data to be stored in an addressed cache data line, said input data including multiple bytes with a least significant byte to be stored in a non-least significant byte location of a first double word in said cache data line and including a most significant byte to be stored in a non-most significant byte location of a second adjacent double word in said cache data line, said input circuitry selectively shifting said input data to position said least significant byte in said non-least significant byte location and to position said most significant byte in said non-most significant byte position;

a data output path coupled to said RAM data outputs, said data output path receiving output data from an enabled cache data line and aligning said output data, said output data including multiple bytes with a least significant byte stored in a non-least significant byte location of a first double word in said cache data line and including a most significant byte stored in a non-most significant byte location of a second adjacent double word in said cache data line, said output circuitry selectively shifting said output data to position said least significant byte to the least significant byte position of said multiple byte output; and

an instruction output path coupled to said RAM data outputs that selects and outputs a plurality of instruction bytes from a plurality of double words of an addressed cache data line.

9. A cache memory system as defined in claim 8, wherein said input circuitry further includes:

an input multiplexer coupled to an internal bus and an external data bus, said input multiplexer selectively communicating said input data from one of said internal bus or said external bus to a multiplexer output; and

an input barrel shifter coupled between said output of said input multiplexer and said RAM data inputs, said input barrel shifter shifts said input data.

10. A cache memory system as defined in claim 8, wherein said data output path includes:

a plurality of data multiplexers coupled to said plurality of RAM data outputs, said plurality of data multiplexers receive and selectively transfer said output data; and

an output barrel shifter coupled to said plurality of data multiplexers, said output barrel shifter receiving and shifting said output data.

11. A cache memory system as defined in claim 8, wherein said instruction output path further includes:

an instruction multiplexer coupled to said plurality of RAM data outputs, said instruction multiplexer selects a plurality of said instruction bytes; and

an instruction register coupled to said instruction multiplexer outputs, said instruction register communicates said plurality of instruction bytes to an instruction bus.

12. A cache memory system as defined in claim 11, wherein said instruction output path accesses said plurality of instruction bytes in a single cycle.

13. A cache memory system as defined in claim 11, wherein said instruction output path accesses a said plurality of instruction bytes in a single cycle and a second plurality of instruction bytes from said enabled cache data line in a second cycle.

14. A method for storing data to a cache memory comprising the steps of:

enabling a cache data line that comprises a plurality of sequential double words; and

aligning input data for storage in said enabled cache data line, said input data to be stored in an addressed cache data line, said input data including multiple bytes with a least significant byte to be stored in a non-least significant byte location of a first double word in said cache data line, and including a most significant byte to be stored in a non-most significant byte

RNSDOCID: <WO 9522791A2 I >

location of a second adjacent double word in said cache data line, said input circuitry selectively shifting said input data to position said least significant byte in said non-least significant byte location and to position said most significant byte in said non-most significant byte position.

15. The method as defined in claim 14 further comprising the step of selecting said input data from a plurality of data buses.

16. The method as defined in claim 14 further comprising the step of providing said input data in sizes of a byte, word or double word.

17. A method for retrieving data from a cache memory comprising the steps of:

enabling a cache data line that comprises a plurality of sequential double words; and

aligning output data from said enabled cache data line, said output data including multiple bytes with a least significant byte stored in a non-least significant byte location of a first double word in said cache data line and including a most significant byte stored in a non-most significant byte location of a second adjacent double word in said cache data line, said output circuitry selectively shifting said output data to position said least significant byte to the least significant byte position of said multiple byte output.

18. The method as defined in claim 17 further comprising the steps of providing said output data in sizes of a byte, word or double word.

19. A method for retrieving data from a cache memory comprising the steps of:

enabling a first cache data line that comprises a plurality of sequential double words;

aligning output data in said enabled cache data line, said output data including multiple bytes with a least significant byte stored in a non-least significant byte location of a first double word in said cache data line and including a most significant byte stored in a non-most significant byte location of a second adjacent double word in said cache data line, said output circuitry selectively shifting said output data to position said least significant byte to the least significant byte position of said multiple byte output;

enabling a second cache data line; and

selecting a plurality of instruction bytes from said enabled second cache data line.

20. The method as defined in claim 19 further comprising the steps of:

BNSDOCID: <WO 9522791A2 I >

forwarding said output data to a local data bus; and

forwarding said output instructions to a local instruction bus.

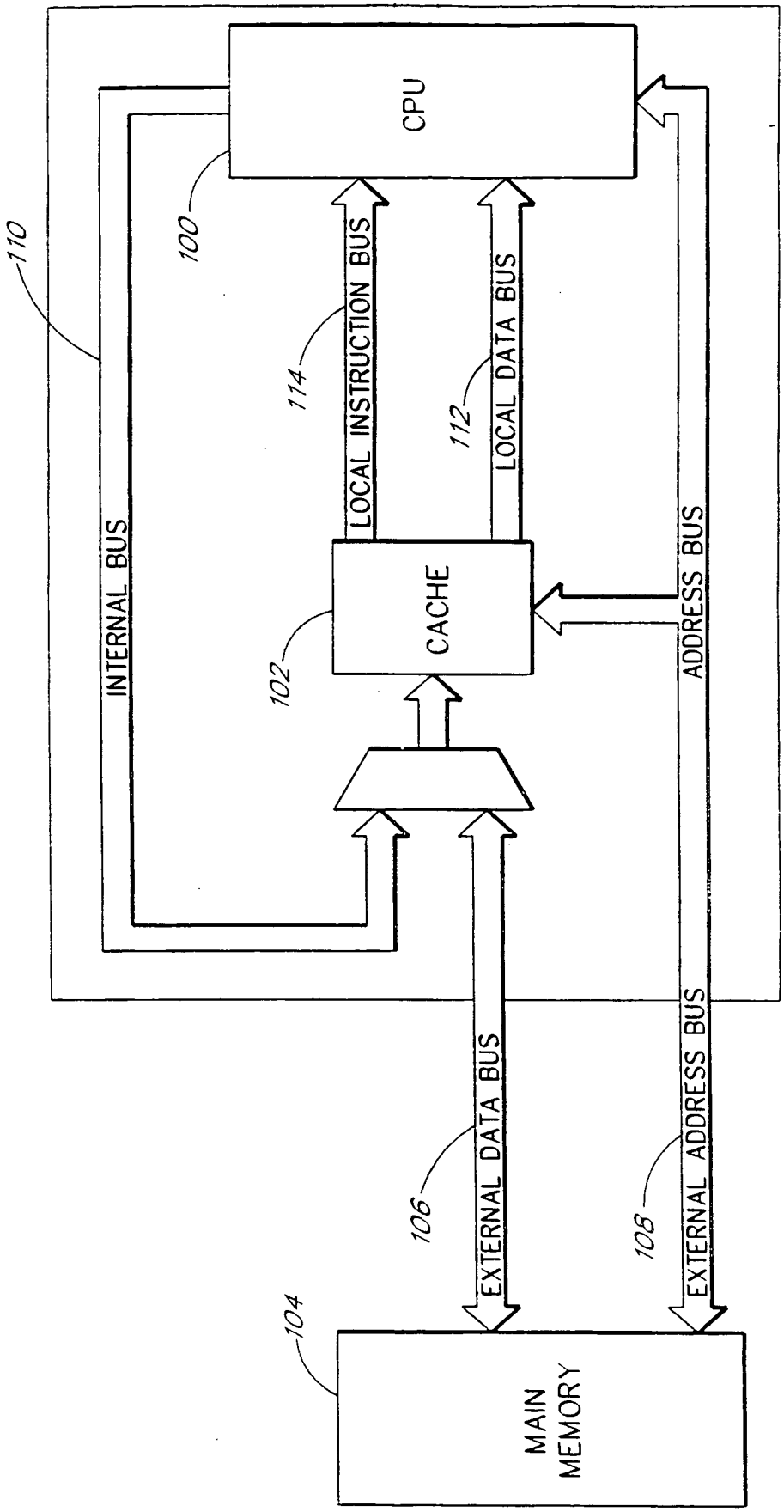


FIG. 1

PNSDOCID: WO 9522791A2 1 -

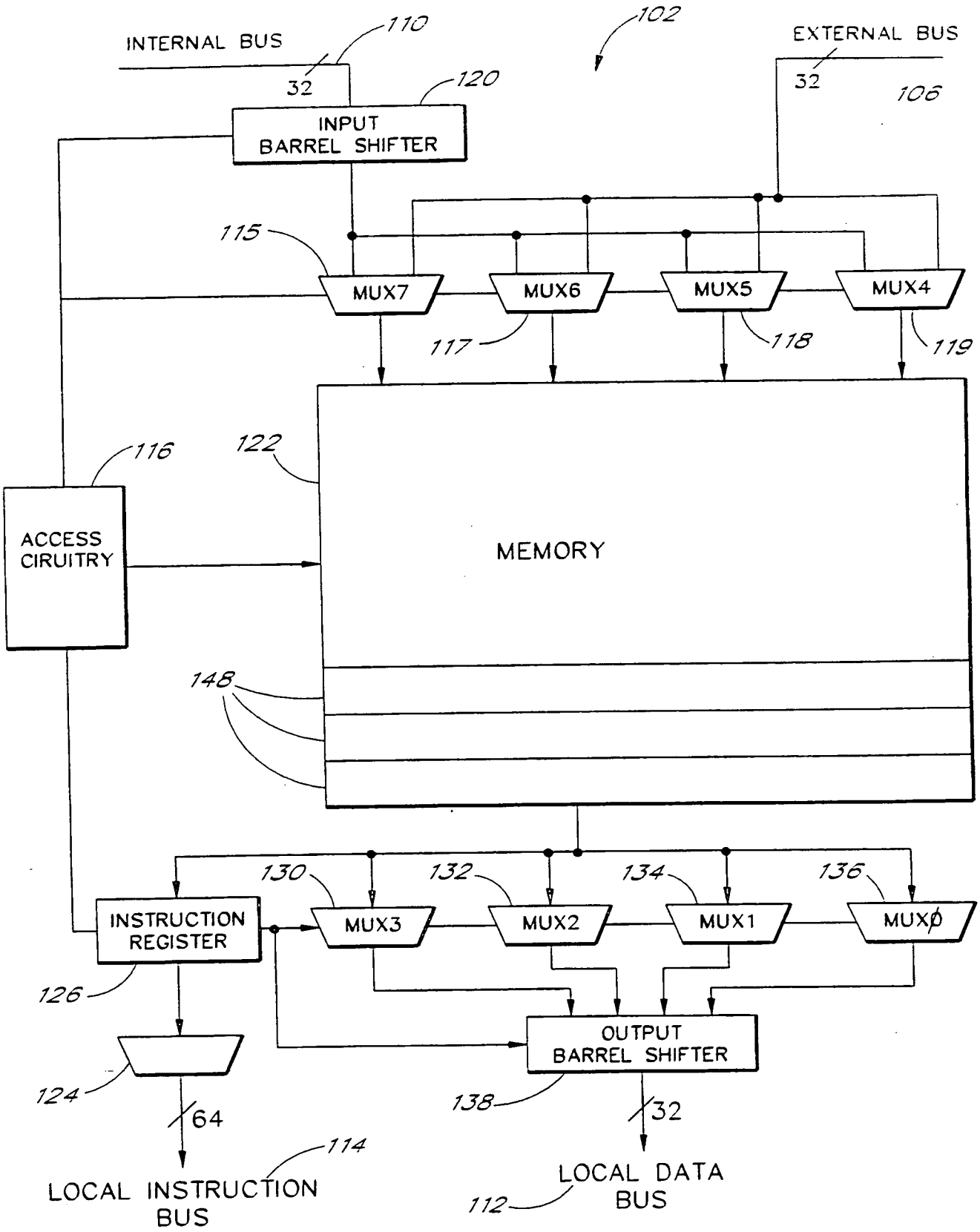

INTERNAL BUS
110
32
102
EXTERNAL BUS
32
106
120
INPUT BARREL SHIFTER
115
MUX7
MUX6
MUX5
MUX4
117
118
119
116
122
ACCESS CIRUITRY
MEMORY
148
130
132
134
136
INSTRUCTION REGISTER
MUX3
MUX2
MUX1
MUXØ
126
OUTPUT BARREL SHIFTER
124
138
64
32
114
LOCAL INSTRUCTION BUS
112
LOCAL DATA BUS

FIG. 2

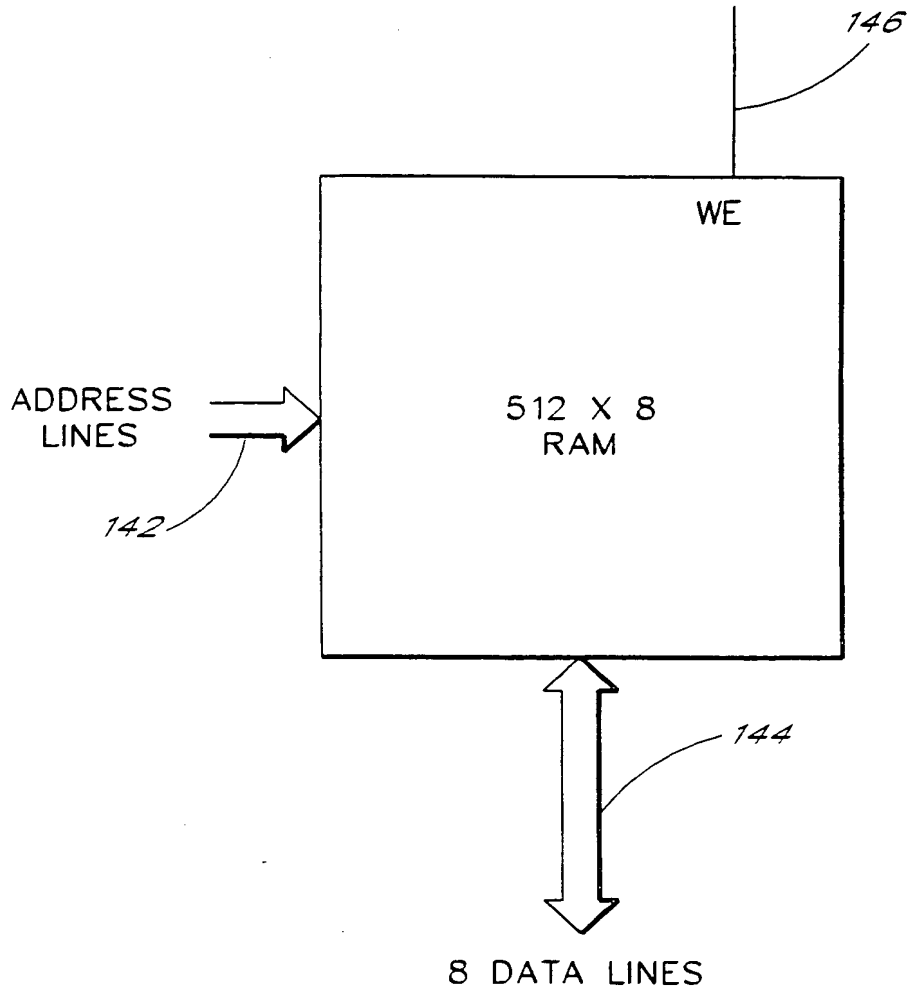


FIG. 3

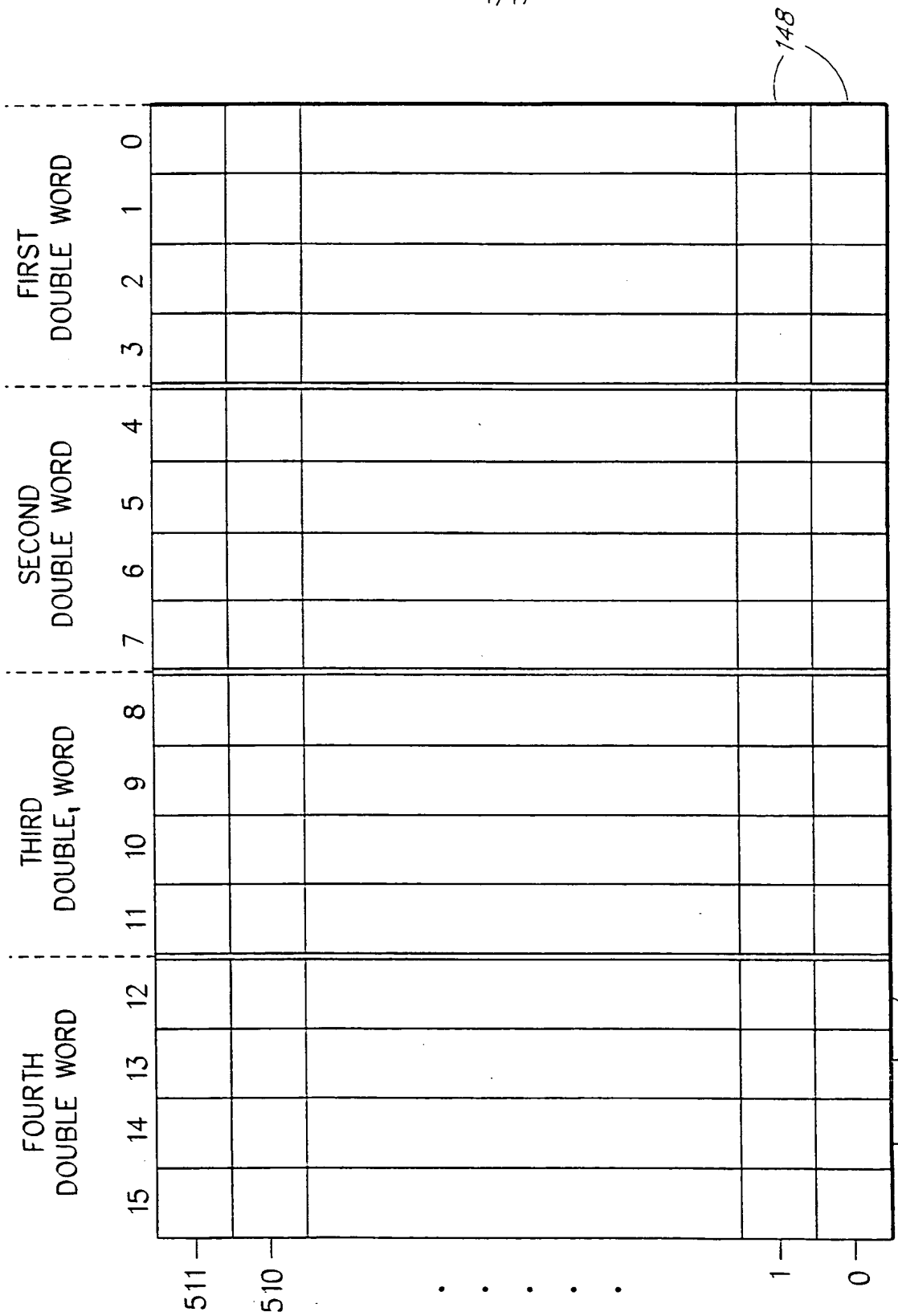

148
FIRST DOUBLE WORD
SECOND DOUBLE WORD
THIRD DOUBLE, WORD
FOURTH DOUBLE WORD
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
511
510
1
0
140

FIG. 4

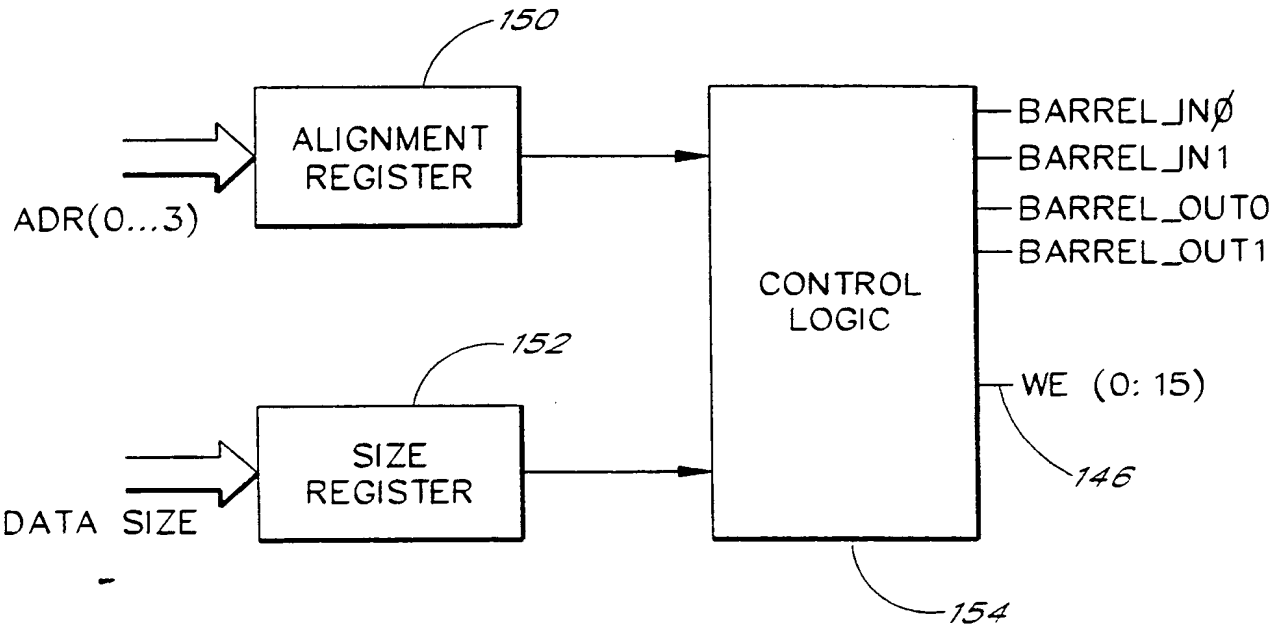

150
ALIGNMENT REGISTER
ADR(0...3)
152
SIZE REGISTER
DATA SIZE
CONTROL LOGIC
BARREL_JNØ
BARREL_JN1
BARREL_OUT0
BARREL_OUT1
WE (0: 15)
146
154

FIG. 5

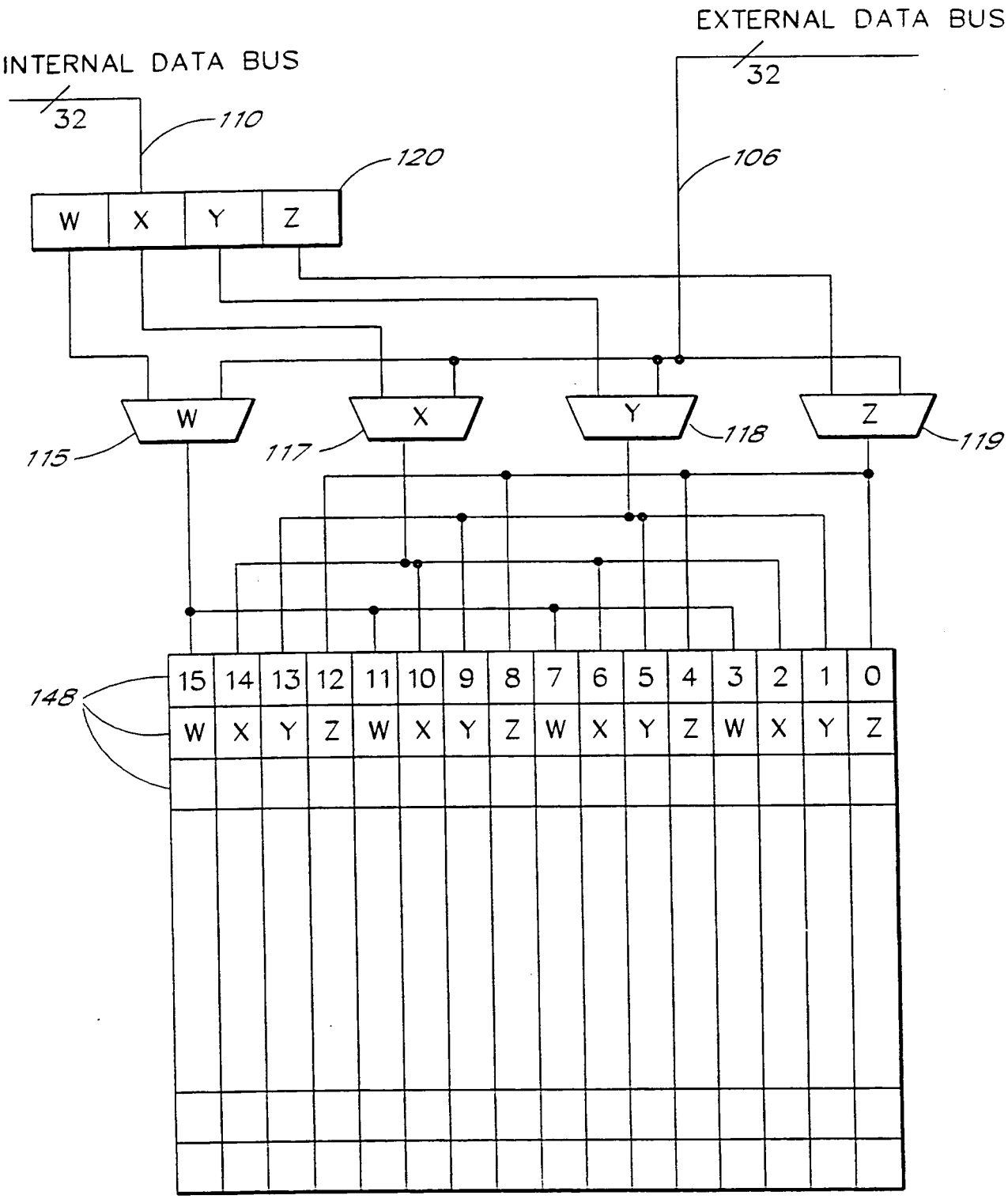


FIG. 6

BNSDOCID: <WO_ 9522791A2 I >

Align Reg.	BARREL_IN1	BARREL_IN0	Command
0	0	0	No shifting
1	0	1	Wrap Around Shift Left 1 Byte
2	1	0	Wrap Around Shift Left 2 Bytes
3	1	1	Wrap Around Shift Left 3 bytes
4	0	0	No Shifting
5	0	1	Wrap Around Shift Left 1 Byte
6	1	0	Wrap Around Shift Left 2 Bytes
7	1	1	Wrap Around Shift Left 3 Bytes
8	0	0	No Shifting
9	0	1	Wrap Around Shift Left 1 Byte
A	1	0	Wrap Around Shift Left 2 Bytes
B	1	1	Wrap Around Shift Left 3 Bytes
C	0	0	No Shifting
D	0	1	Wrap Around Shift Left 1 Byte Only Write Portion Within Cache Boundary
E	1	0	Wrap Around Shift Left 2 Bytes Only Write Portion Within Cache Boundary
F	1	1	Wrap Around Shift Left 3 Bytes Only Write Portion Within Cache Boundary

FIG. 7

Alignment (Hex)	Size	Fourth Double Word 15 14 13 12	Third Double Word 11 10 9 8	Second Double Word 7 6 5 4	First Double Word 3 2 1 0
0	BYTE				X
0	WORD				X X
0	DWORD				X X X X
1	BYTE				X
1	WORD				X X
1	DWORD			X	X X X
2	BYTE				X
2	WORD				X X
2	DWORD			X X	X X
3	BYTE				X
3	WORD			X	X
3	DWORD			X X X	X
4	BYTE			X	
4	WORD			X X	
4	DWORD			X X X X	
5	BYTE			X	
5	WORD			X X	
5	DWORD		X	X X X	
6	BYTE			X	
6	WORD			X X	
6	DWORD		X X	X X	
7	BYTE			X	
7	WORD		X	X	
7	DWORD		X X X	X	

FIG. 8A

PNSDOCID: <WO 9522791A2 1 >

Alignment (Hex)	Size	Fourth Double Word 15 14 13 12	Third Double Word 11 10 9 8	Second Double Word 7 6 5 4	First Double Word 3 2 1 0
8	BYTE		X		
8	WORD		X X		
8	DWORD		X X X X		
9	BYTE		X		
9	WORD		X X		
9	DWORD	X	X X X		
A	BYTE		X		
A	WORD		X X		
A	DWORD	X X	X X		
B	BYTE		X		
B	WORD	X	X		
B	DWORD	X X X	X		
C	BYTE	X			
C	WORD	X X			
C	DWORD	X X X X			
D	BYTE	X			
D	WORD	X X			
E	BYTE	X			
E	WORD	X X			
F	BYTE	X			

FIG. 8B

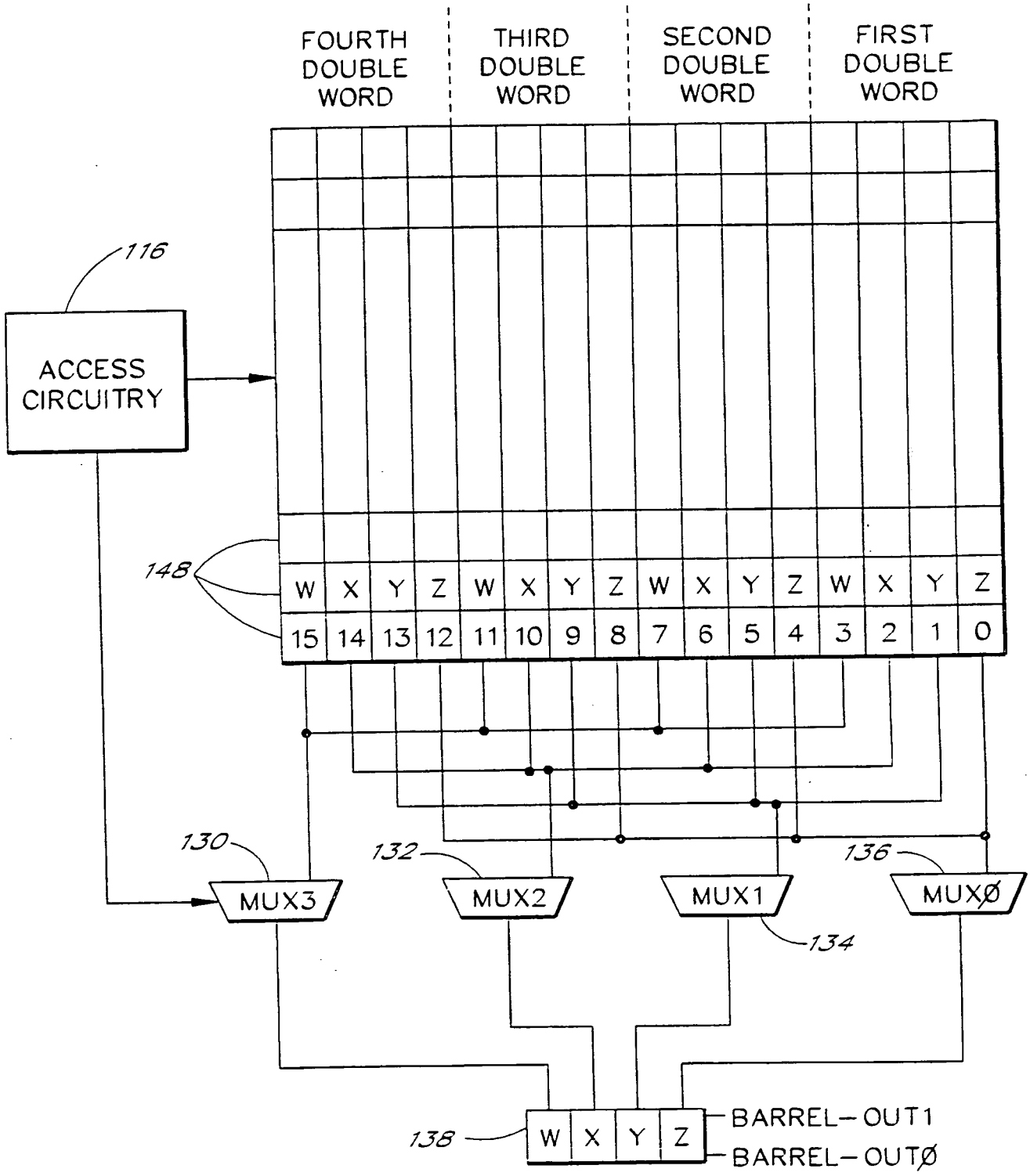


FIG. 9

BNSDOCID: <WO 9522791A2 1 >

Align Reg.	Bytes to Retrieve	MUX3 Output	MUX2 Output	MUX1 Output	MUX0 Output
0	3, 2, 1, 0	3	2	1	0
1	4, 3, 2, 1	3	2	1	4
2	5, 4, 3, 2	3	2	5	4
3	6, 5, 4, 3	3	6	5	4
4	7, 6, 5, 4	7	6	5	4
5	8, 7, 6, 5	7	6	5	8
6	9, 8, 7, 6	7	6	9	8
7	10, 9, 8, 7	7	10	9	8
8	11,10, 9, 8	11	10	9	8
9	12,11,10, 9	11	10	9	12
A	13,12,11,10	11	10	13	12
B	14,13,12,11	11	14	13	12
C	15,14,13,12	15	14	13	12
D	15,14,13	15	14	13	X
E	15,14	15	14	X	X
F	15	15	X	X	X

FIG. 10

BNSDOCID: <WO 9522791A2 I >

Align Reg.	BARREL_OUT1	BARREL_OUT0	Command
0	0	0	No shifting
1	0	1	Wrap Around Shift Right 1 Byte
2	1	0	Wrap Around Shift Right 2 Bytes
3	1	1	Wrap Around Shift Right 3 Bytes
4	0	0	No shifting
5	0	1	Wrap Around Shift Right 1 Byte
6	1	0	Wrap Around Shift Right 2 Bytes
7	1	1	Wrap Around Shift Right 3 Bytes
8	0	0	No Shifting
9	0	1	Wrap Around Shift Right 1 Byte
A	1	0	Wrap Around Shift Right 2 Bytes
B	1	1	Wrap Around Shift Right 3 Bytes
C	0	0	No Shifting
D	0	0	No Shifting
E	0	0	No Shifting
F	0	0	No Shifting

FIG. 11

BNSDOCID: <WO 9522791A2 I >

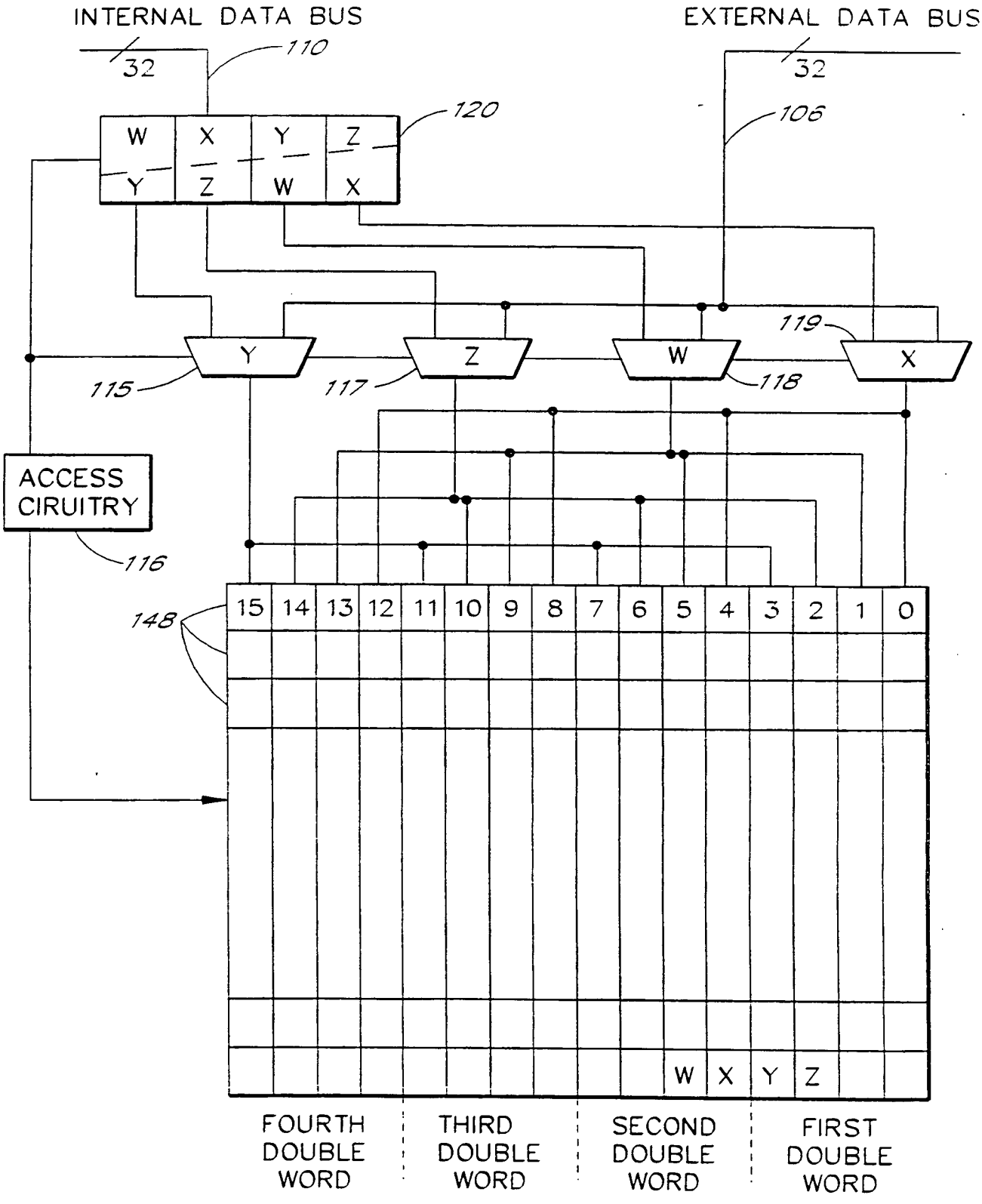

INTERNAL DATA BUS
32
110
120
W X Y Z
Y Z W X
EXTERNAL DATA BUS
32
106
119
Y
Z
W
X
115
117
118
ACCESS CIRUITRY
116
148
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
W X Y Z
FOURTH DOUBLE WORD
THIRD DOUBLE WORD
SECOND DOUBLE WORD
FIRST DOUBLE WORD
158
156
W X
Y Z
MAIN MEMORY

FIG. 12

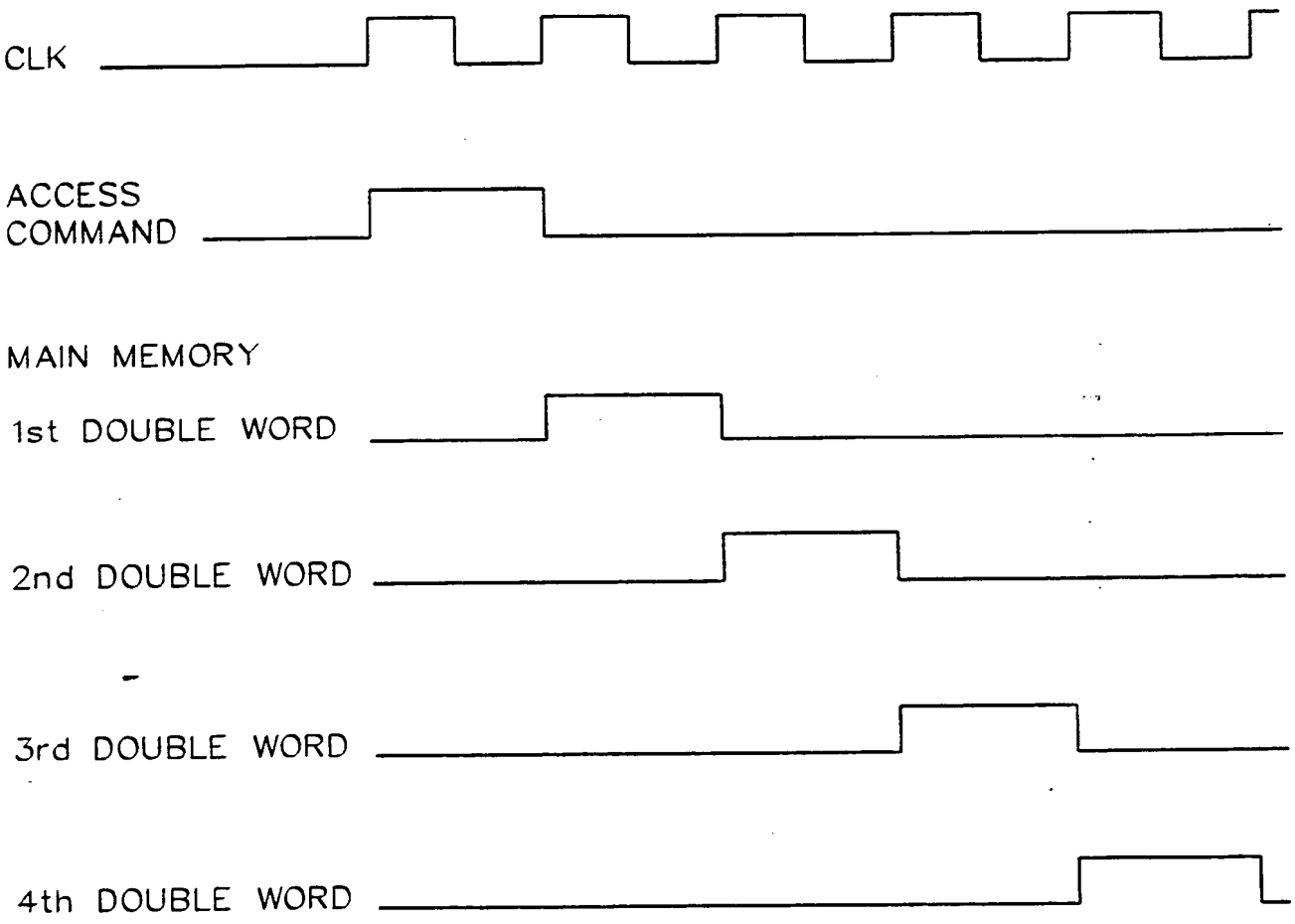


FIG. 13

BNSDOCID: <WO__9522791A2 I_>

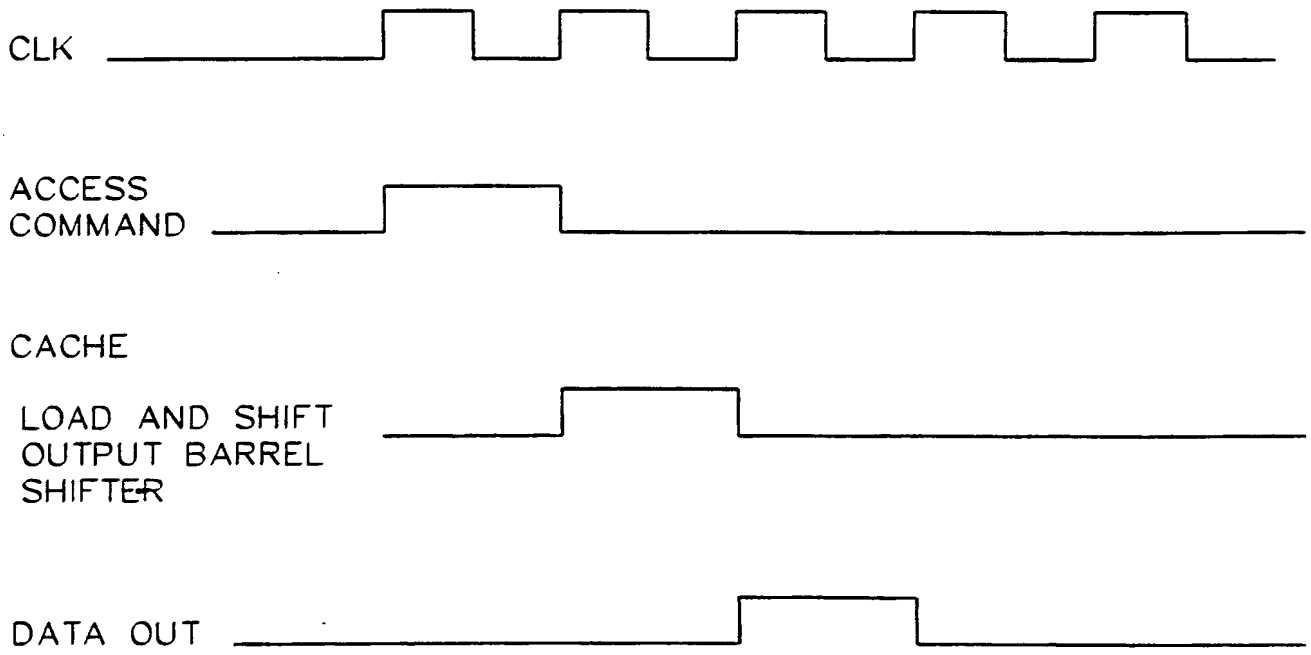

CLK
ACCESS COMMAND
CACHE
LOAD AND SHIFT OUTPUT BARREL SHIFTER
DATA OUT

FIG. 14

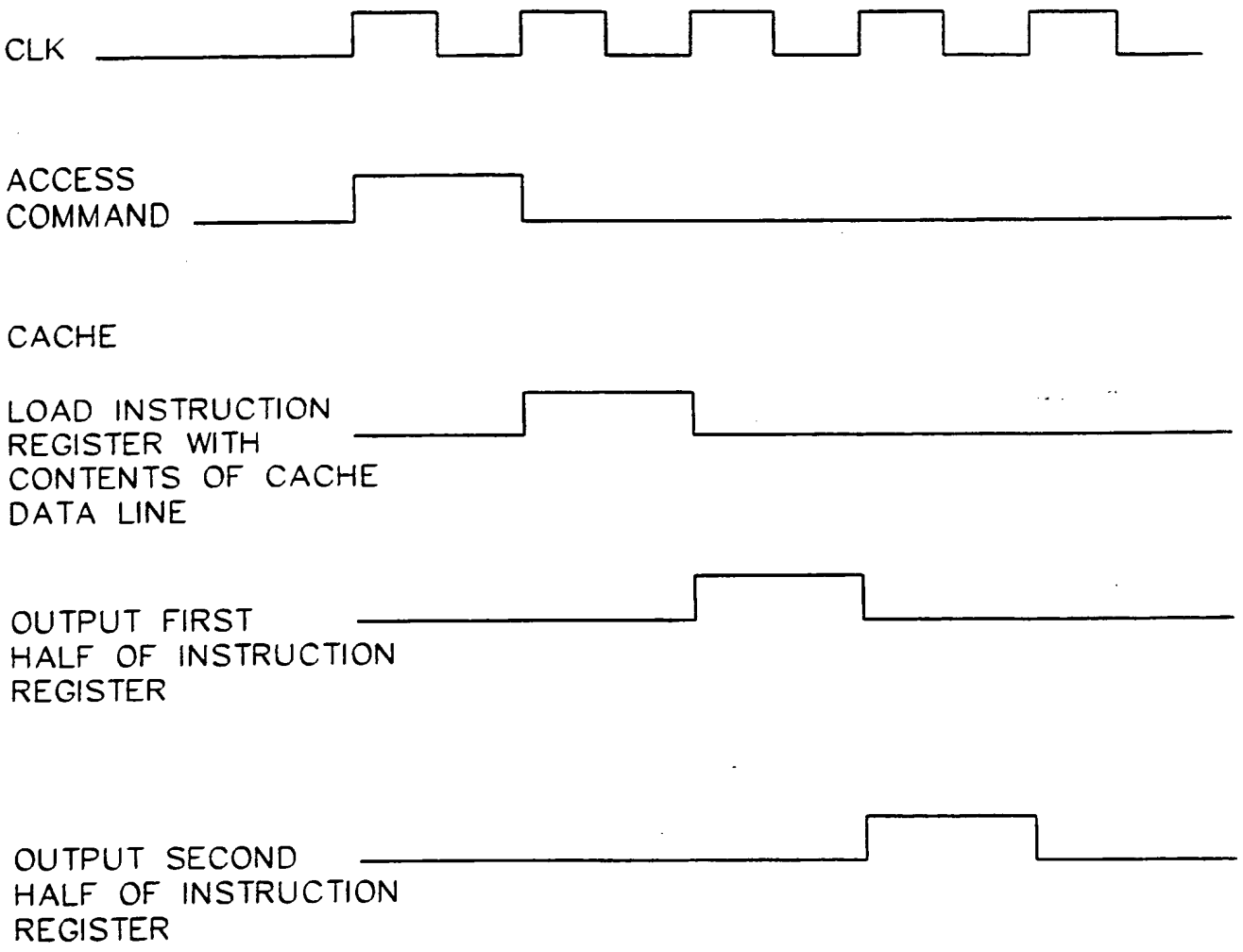

CLK
ACCESS COMMAND
CACHE
LOAD INSTRUCTION REGISTER WITH CONTENTS OF CACHE DATA LINE
OUTPUT FIRST HALF OF INSTRUCTION REGISTER
OUTPUT SECOND HALF OF INSTRUCTION REGISTER

FIG. 15

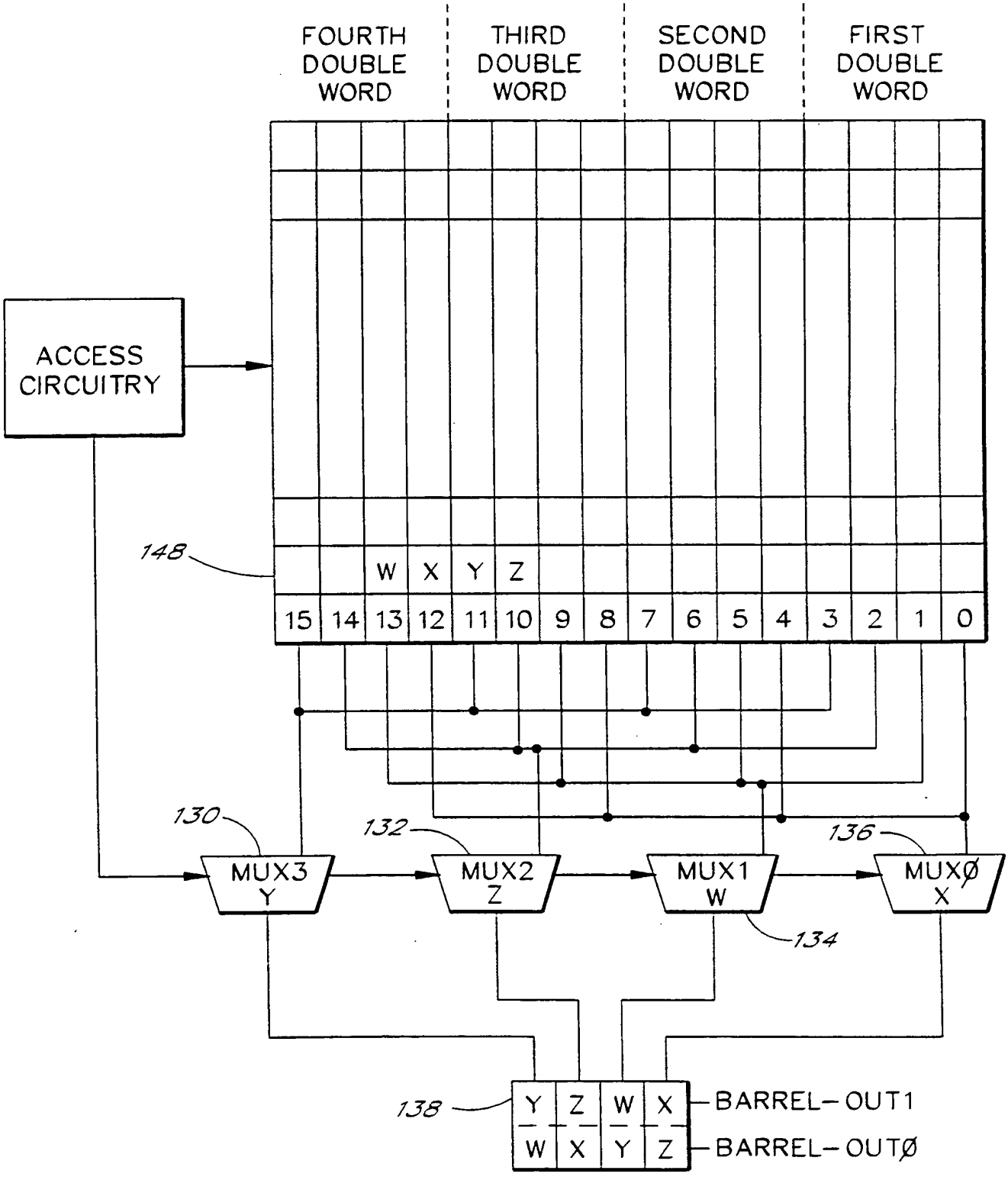

FOURTH DOUBLE WORD
THIRD DOUBLE WORD
SECOND DOUBLE WORD
FIRST DOUBLE WORD
ACCESS CIRCUITRY
148
W X Y Z
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
130
132
136
MUX3 Y
MUX2 Z
MUX1 W
MUXØ X
134
138
Y Z W X
W X Y Z
BARREL-OUT1
BARREL-OUTØ

FIG. 16

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 :
G06F 13/00

A3

(11) International Publication Number: WO 95/22791

(43) International Publication Date: 24 August 1995 (24.08.95)

(21) International Application Number: PCT/US95/01779

(22) International Filing Date: 8 February 1995 (08.02.95)

(30) Priority Data:
08/193,383 8 February 1994 (08.02.94) US

(71) Applicant: MERIDIAN SEMICONDUCTOR, INC. [US/US]; Suite 145, 17310 Redhill Avenue, Irvine, CA 92714 (US).

(72) Inventors: WHITTED, Graham, B., III; 25 Hermosa, Irvine, CA 92720 (US). KANE, James, A.; 114 Via Orvieto, Newport Beach, CA 92663 (US). CHANG, Hsiao-Shih; 1134 North Palo Loma Place, Orange, CA 92669 (US).

(74) Agent: SEWELL, Jerry, T.; Knobbe, Martens, Olson and Bear, 16th floor, 620 Newport Center Drive, Newport Beach, CA 92660 (US).

(81) Designated States: CN, JP, KR, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

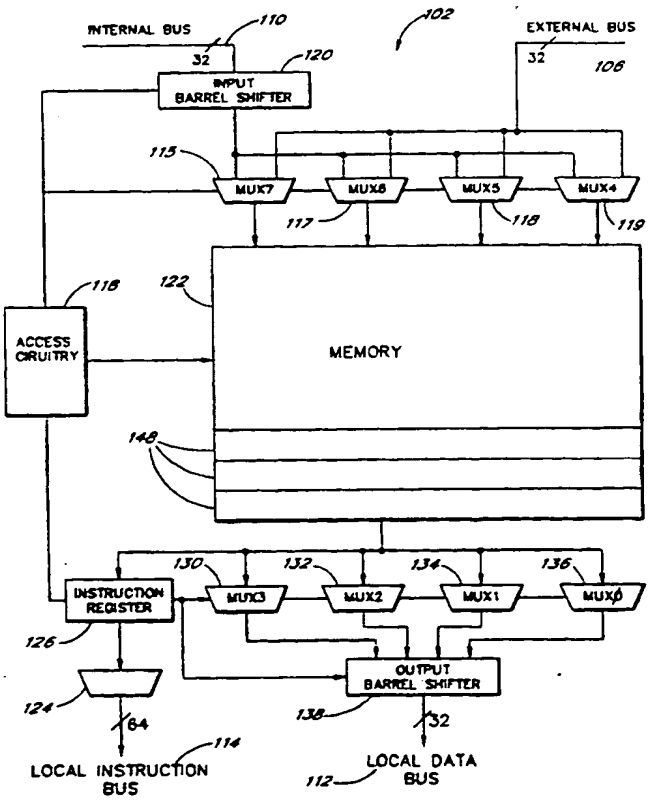
Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(88) Date of publication of the international search report:
21 September 1995 (21.09.95)

(54) Title: METHOD AND APPARATUS FOR SINGLE CYCLE CACHE ACCESS ON DOUBLE WORD BOUNDARY CROSS

(57) Abstract

A cache memory system and method control a random access read/write cache memory (122) that stores a plurality of cache data lines (148). The bytes in each cache data line (148) correspond to bytes stored in sequential addresses in a main memory (104). The cache memory (122) is organized to provide storage of a plurality of double words in each cache data line (148). To retain main memory consistency, input circuitry receives and ligns data before storage in the cache memory (122). A separate instruction output path (114) provides access to a plurality of double words in a single cycle. A separate data output path (112) provides access to data that crosses a double word boundary in a single cycle and provides the data as aligned data.



SNSDOCID: <WO 9522791A1 >

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	**GB**	United Kingdom	**MR**	Mauritania
AU	Australia	**GE**	Georgia	**MW**	Malawi
BB	Barbados	**GN**	Guinea	**NE**	Niger
BE	Belgium	**GR**	Greece	**NL**	Netherlands
BF	Burkina Faso	**HU**	Hungary	**NO**	Norway
BG	Bulgaria	**IE**	Ireland	**NZ**	New Zealand
BJ	Benin	**IT**	Italy	**PL**	Poland
BR	Brazil	**JP**	Japan	**PT**	Portugal
BY	Belarus	**KE**	Kenya	**RO**	Romania
CA	Canada	**KG**	Kyrgyzstan	**RU**	Russian Federation
CF	Central African Republic	**KP**	Democratic People's Republic of Korea	**SD**	Sudan
CG	Congo			**SE**	Sweden
CH	Switzerland	**KR**	Republic of Korea	**SI**	Slovenia
CI	Côte d'Ivoire	**KZ**	Kazakhstan	**SK**	Slovakia
CM	Cameroon	**LI**	Liechtenstein	**SN**	Senegal
CN	China	**LK**	Sri Lanka	**TD**	Chad
CS	Czechoslovakia	**LU**	Luxembourg	**TG**	Togo
CZ	Czech Republic	**LV**	Latvia	**TJ**	Tajikistan
DE	Germany	**MC**	Monaco	**TT**	Trinidad and Tobago
DK	Denmark	**MD**	Republic of Moldova	**UA**	Ukraine
ES	Spain	**MG**	Madagascar	**US**	United States of America
FI	Finland	**ML**	Mali	**UZ**	Uzbekistan
FR	France	**MN**	Mongolia	**VN**	Viet Nam
GA	Gabon				

RNSDOCID: <WO 9523791A2 I >

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US95/01779

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :G06F 13/00
US CL : 395/425

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 395/425, 445, 411, 421.08, 421.09

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y,P	US, A, 5,386,531 (Blaner et al.) 31 JANUARY 1995, (claims 1, 14 and 17); (column 9, lines 34-57); (column 10, lines 6-22); (Fig. 2B, WROTATE SHIFTER); (FIG. 2B, MUX); (column 7, line 65 - column 8, line 10); (Fig. 2B, RMERGE);	1-20
A	US, A, 4,814,976 (Hansen et al) 21 MARCH 1989, see abstract.	1-20
A	US, A, 4,314,332 (Shiraogawa et al.) 02 FEBRUARY 1982, see abstract.	1-20
A	US, A, 4,652,991 (Yamano) 24 MARCH 1987, see abstract.	1-20

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be part of particular relevance

"E" earlier document published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
25 JULY 1995	09AUG 1995

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Authorized officer
DAVID J. O'NEILL
Telephone No. (703) 308-5466

ENSDOCID: <WO 9522791A1 I >

THIS PAGE BLANK (USPTO)